

An Entropy-Guided Dual-Decomposition GRU Framework for Bitcoin Price Forecasting Using CEEMDAN and STL

Hoa Tran Thai¹ and Khang Pham-Quoc¹

¹ University of Economics, Hue University,
99 Ho Duc Di Street, Hue City, Vietnam
pqkhang@hueuni.edu.vn

Abstract. Bitcoin price forecasting is complicated by nonlinear dynamics and fluctuations across temporal scales. We propose an entropy-guided framework combining complete ensemble empirical mode decomposition with adaptive noise (CEEMDAN), seasonal-trend decomposition using LOESS (STL), and gated recurrent units (GRUs). CEEMDAN produces nine retained components; sample entropy selects IMF5 for STL refinement, yielding eleven component-specific forecasts that are additively reconstructed. The reported experiment uses 4,102 daily BTC-USD observations from 2015–2026 and compares GRU, STL–GRU, and CEEMDAN–GRU. The proposed model attains RMSE of USD 2,020.99, MAE of USD 1,674.40, MAPE of 2.0351%, and R^2 of 0.9911. RMSE is 33.53% lower than GRU and 10.39% lower than STL–GRU. These single-split results motivate further evaluation; they do not establish general superiority or prospective validity without verification of causal preprocessing.

Keywords: Bitcoin price forecasting, CEEMDAN, GRU, STL, dual decomposition.

1 Introduction

Bitcoin price forecasting has attracted research on statistical, machine-learning, and recurrent neural models. Comparative studies show that conclusions depend on the dataset, input variables, evaluation period, and model configuration [1], [2]. Accurate price-level prediction therefore requires a clearly defined information set and comparison protocol; a low forecast error alone does not establish trading profitability.

Decomposition-based forecasting separates a price series into components that can be modeled individually. Li et al. [3] combined variational mode decomposition (VMD) with bidirectional long short-term memory networks for Bitcoin prediction. Chang et al. [4] grouped CEEMDAN components by frequency, while Dong and Zhou [5] used sample entropy to reconstruct components before hybrid forecasting. These studies motivate adapting the downstream model to component characteristics rather than treating every decomposed signal identically.

We investigate a selective refinement strategy: CEEMDAN provides an initial multiscale representation, sample entropy identifies the most irregular retained component, and STL refines that component into trend, seasonal, and remainder terms. Separate GRUs forecast the resulting signals, whose predictions are summed on the original price scale. The design tests whether local refinement of one component can

improve a CEEMDAN–GRU model without applying a second decomposition to every component.

The contribution is a specified component-selection and reconstruction mechanism, evaluated against raw-series GRU and two single-stage decomposition variants. The empirical scope is one asset, one horizon, and one reported chronological split. We quantify gains over each baseline and discuss the additional component-model count, while distinguishing the observed ranking from untested claims about decomposition choice, statistical significance, or deployment performance.

2 Related Work

2.1 Recurrent Models and Decomposition-Based Forecasting

Recurrent architectures learn dependencies in sequential inputs. GRU uses reset and update gates to control the candidate state and the retained information [6]. Its simpler gating structure motivates its use as the common learner in this study; this architectural choice does not imply that it is uniformly more accurate or faster than LSTM. Cryptocurrency comparisons report results that vary with architecture and asset [1], [2].

Decomposition offers a complementary representation of the forecasting inputs. The VMD-based study of Li et al. [3] demonstrates that CEEMDAN is not the only plausible initial method. Chang et al. [4] combine CEEMDAN with clustering and IMF reconstruction for Bitcoin, and Dong and Zhou [5] combine CEEMDAN, sample entropy, ARIMA, CNN, and LSTM for financial forecasting. Their use of component grouping or reconstruction supports examining heterogeneity among modes, but their reported errors are not directly comparable with ours because the datasets and evaluation settings differ.

2.2 Scope of the Selective Refinement Strategy

Our framework preserves all retained CEEMDAN outputs except the component with the highest sample entropy, which is replaced by three STL terms. Thus, the distinction is the selective CEEMDAN-to-STL transformation and its integration with a common GRU learner. We do not claim that multistage decomposition or entropy-based processing is new in general. The four-model comparison evaluates the combined design; it does not isolate entropy selection from an alternative component-selection rule.

3 Proposed CEEMDAN–STL–GRU Methodology

3.1 Why CEEMDAN Is Used First

CEEMDAN is selected for its adaptive mode extraction and additive reconstruction, both of which suit the proposed component-wise forecasting architecture [7]. Unlike EMD [8], which can mix oscillations of different scales within one intrinsic mode

function (IMF), CEEMDAN averages noise-assisted extractions while updating a common residual. Relative to EEMD [9], the stage-wise residual construction addresses finite-ensemble reconstruction inconsistencies. This motivates CEEMDAN as a representation method; it does not guarantee noise-free components or smaller forecasting errors.

VMD is a strong alternative because it jointly estimates band-limited modes [10]. However, its mode count and bandwidth penalty require specification or tuning. We favor CEEMDAN here because its sequential extraction does not require a fixed wavelet basis or an a priori number of modes, although stopping rules and noise parameters remain necessary. Wavelet decomposition provides an efficient multiresolution representation [11], but requires a wavelet family, decomposition level, and boundary convention. CEEMDAN instead derives modes from local extrema, allowing the entropy rule to operate on data-adaptive components.

Table 1 summarizes the design trade-offs. CEEMDAN still depends on ensemble size, noise amplitude, interpolation, and endpoint treatment; empirical-mode methods can produce artifacts when these choices are inappropriate [12]. Accordingly, we justify suitability rather than empirical superiority over EMD, EEMD, VMD, or wavelets. The present experiment does not compare those initial decompositions under a common tuning budget.

Table 1. Rationale for the initial decomposition; no cross-method benchmark is implied.

Method	Relevant property	Implication for this framework
EMD [8]	Adaptive extraction from local extrema	Mode mixing can weaken component separation.
EEMD [9]	Noise-assisted ensemble averaging	Finite ensembles may retain noise and reconstruction discrepancies.
CEEMDAN [7]	Adaptive extraction with recursive residuals	Supports additive reconstruction and entropy-based selection; parameters still matter.
VMD [10]	Joint estimation of band-limited modes	Requires the mode count and bandwidth penalty to be set or tuned.
Wavelets [11]	Multiresolution filtering	Requires a basis, level, and boundary convention.

3.2 Forecasting Task and CEEMDAN Representation

Let p_t denote the daily BTC-USD closing price. With lookback $L = 30$, the objective is a one-day-ahead price-level forecast:

$$\mathbf{x}_t = (p_{t-L+1}, \dots, p_t)^T, \quad \hat{p}_{t+1} = f_\theta(\mathbf{x}_t). \quad (1)$$

Figure 1 shows the component flow for the reported nine-output decomposition. These output counts describe this experiment; CEEMDAN can return a different number for another series or observation window.

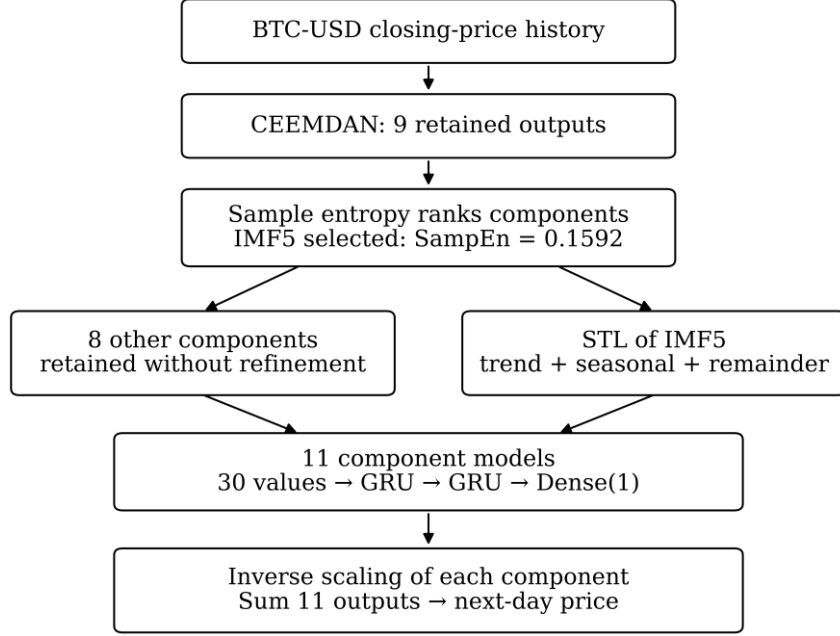


Fig. 1. Selective refinement and additive prediction flow. The 9-to-11 component example is the reported configuration; prospective preprocessing requirements are discussed in Section 3.5.

Let $E_k(\cdot)$ denote the k th EMD mode, I the ensemble size, $w^{(i)}$ independent unit-variance white-noise realizations, and ε_k the noise amplitude at stage k . CEEMDAN begins with [7]

$$c_1 = \frac{1}{I} \sum_{i=1}^I E_1[p + \varepsilon_0 w^{(i)}], \quad r_1 = p - c_1. \quad (2)$$

For $k \geq 2$, it extracts a mode from each perturbed residual and updates the common residual:

$$c_k = \frac{1}{I} \sum_{i=1}^I E_1[r_{k-1} + \varepsilon_{k-1} E_{k-1}(w^{(i)})] . \quad (3)$$

$$r_k = r_{k-1} - c_k.$$

Extraction stops when the residual no longer supports another mode. Including the terminal residual preserves additive reconstruction. The retained collection C contains J components, counting the residual once:

$$p = \sum_{k=1}^K c_k + r_K = \sum_{j=1}^J c_j, \quad J = 9. \quad (4)$$

The final retained output need not be an oscillatory IMF. Using “retained components” avoids conflating an IMF with the terminal trend or residual. Lower-order modes commonly contain faster oscillations, but frequency order alone is not a measure of forecasting difficulty.

3.3 Entropy Selection and STL Refinement

Sample entropy measures pattern irregularity while excluding self-matches [13]. For each component c_j , compare distinct length- m template pairs using their maximum absolute coordinate-wise difference. Let $B_m(\rho)$ count pairs within tolerance ρ and $A_{m+1}(\rho)$ count the same pairs that also match after extension to length $m + 1$. Both counts use the common eligible starting indices $1, \dots, N - m$. Then

$$\text{SampEn}(m, \rho) = -\ln \frac{A_{m+1}(\rho)}{B_m(\rho)}. \quad (5)$$

The comparison requires a consistent embedding dimension, tolerance convention, and eligible sample across components. If either match count is zero, the estimate is not finite and a predefined handling rule is required. The selection rule is

$$j^* = \arg \max_{1 \leq j \leq J} \text{SampEn}(c_j). \quad (6)$$

In the reported calculation, IMF5 has the largest sample entropy, 0.1592, and is selected for refinement. This value is specific to the entropy settings and scale convention. High sample entropy does not, by itself, identify high-frequency noise, demonstrate that a component is unforecastable, or show that STL will improve it.

STL separates the selected component into a trend τ_t , a seasonal term s_t at a specified period, and a remainder e_t [14]:

$$c_{j^*,t} = \tau_t + s_t + e_t. \quad (7)$$

LOESS smoothing estimates the trend and seasonal terms; an optional robustness loop downweights extreme residuals. Here, the seasonal term denotes variation at the chosen STL period, not established calendar seasonality in Bitcoin. The remainder is retained and forecast, so the procedure reorganizes the signal rather than discarding noise. Replacing one component with three gives

$$D = \{c_j : j \neq j^*\} \cup \{\tau, s, e\}, \quad M = 11. \quad (8)$$

For example, the reported decomposition produces eight unchanged CEEMDAN components plus the three STL outputs of IMF5. Direct STL–GRU instead forecasts three terms from the original price series, whereas CEEMDAN–GRU forecasts the nine retained CEEMDAN components without refinement.

3.4 Component GRUs and Additive Reconstruction

Each component is modeled independently with a two-layer GRU followed by a scalar dense output. Omitting the component index for clarity, the reset gate a_t , update gate z_t , candidate state $\tilde{h}_t$, and hidden state h_t follow [6]

$$\begin{aligned} a_t &= \sigma(W_a x_t + U_a h_{t-1} + b_a), \\ z_t &= \sigma(W_z x_t + U_z h_{t-1} + b_z), \\ \tilde{h}_t &= \tanh(W_h x_t + U_h (a_t \odot h_{t-1}) + b_h), \\ h_t &= (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t. \end{aligned} \quad (9)$$

Here σ is the sigmoid function, $\odot$ denotes element-wise multiplication, and W , U , and b are trainable parameters. In this convention, z_t weights the candidate state. The first GRU returns a sequence, the second returns a final representation, and dropout follows each recurrent layer. For component d_m , a 30-value window yields a scalar prediction; the final price is

$$\begin{aligned} \hat{d}_{m,t+1} &= g_{\phi_m}(d_{m,t-29}, \dots, d_{m,t}), \\ \hat{p}_{t+1} &= \sum_{m=1}^M \hat{d}_{m,t+1}, \quad M = 11. \end{aligned} \quad (10)$$

The summands in (10) must be in the same USD scale. If each component has its own MinMax scaler, each prediction is inverse-transformed with that component's scaler before summation. Adding independently normalized predictions would violate the reconstruction identity.

Each GRU minimizes component-level mean squared error over the N_{tr} training targets indexed by T_{tr} :

$$L_m = \frac{1}{N_{tr}} \sum_{t \in T_{tr}} (d_{m,t+1} - \hat{d}_{m,t+1})^2. \quad (11)$$

3.5 Temporal Partition and Preprocessing Requirements

The reported protocol reserves the final 20% of observations for testing. Within the preceding 80%, the last 20% is used for validation, corresponding approximately to 64% training, 16% validation, and 20% testing before window construction. Target dates determine split membership; an input window may include earlier observations, but never observations after its forecast origin. The reported optimizer is Adam, with batch size 32, at most 200 epochs, and early-stopping patience of 20 epochs based on validation loss.

A chronological split alone does not establish an out-of-sample information boundary. For prospective forecasting at origin t , CEEMDAN, entropy selection, STL, and input construction must use only observations available by t ; scalars and tuning choices must be fitted within the development period. Applying decomposition to the complete series before splitting can leak future information through the components. The recorded experiment does not document an origin-by-origin decomposition schedule, so prospective validity of the reported errors remains unverified.

A reproducible causal implementation also needs a rule for matching modes across windows, because their number and ordering can change. Neither that rule nor a new rolling-origin evaluation is inferred from the aggregate scores reported here. These unresolved implementation details are included among the study limitations in Section 4.4.

3.6 Evaluation Metrics

For n evaluation targets, let y_i and $\hat{y}_i$ denote observed and predicted USD prices, and let $\bar{y}$ be the mean of the observed evaluation prices. We use complementary magnitude and relative-error measures [15]:

$$\begin{aligned} \text{MSE} &= \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2, \quad \text{RMSE} = \sqrt{\text{MSE}}, \\ \text{MAE} &= \frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i|, \\ \text{MAPE} &= \frac{100}{n} \sum_{i=1}^n \left| \frac{y_i - \hat{y}_i}{y_i} \right|, \\ R^2 &= 1 - \frac{\sum_{i=1}^n (y_i - \hat{y}_i)^2}{\sum_{i=1}^n (y_i - \bar{y})^2}. \end{aligned} \tag{12}$$

RMSE and MAE are in USD, MSE in USD², and MAPE in percent. MAPE requires nonzero observations; R^2 requires nonzero evaluation variance. A high price-level R^2 does not establish directional accuracy, excess returns, or improvement over a persistence forecast. Comparisons require identical target dates and consistent inverse transformation.

4 Experimental Results and Discussion

4.1 Data and Reported Configuration

The experiment uses 4,102 daily BTC-USD observations from Yahoo Finance over 2015–2026, with closing price as the univariate target. Table 2 consolidates the recorded settings. The comparison comprises a raw-series GRU, STL–GRU, CEEMDAN–GRU, and the proposed CEEMDAN–STL–GRU, each using a 30-day lookahead and a one-day forecast horizon.

Table 2. Recorded experiment settings and component-model counts.

Item	Reported setting
Target and sample	Daily BTC-USD close; 4,102 observations; 2015–2026
Partition	Chronological 80% development / 20% test; last 20% of development for validation
Input and horizon	30 preceding observations; one day ahead

Preprocessing	MinMax scaling; decomposition according to model variant
Learner and training	Two GRU layers; dropout after each; dense scalar output; Adam; batch 32; maximum 200 epochs; patience 20
Selective refinement	9 retained CEEMDAN outputs; IMF5 selected; SampEn = 0.1592; STL replaces it with 3 terms
GRU model counts	GRU: 1; STL-GRU: 3; CEEMDAN-GRU: 9; proposed: 11

Exact date boundaries, post-window sample counts, decomposition parameters, entropy settings, STL period, GRU widths, dropout rates, learning rate, and random seeds are not available in the recorded configuration. Consequently, the study cannot yet be reproduced exactly from these settings alone. This limitation is separate from the arithmetic consistency of the aggregate results.

4.2 Forecasting Performance

Table 3 retains the numerical results of the reported experiment. The proposed framework has the lowest error values and the highest R^2 among the four models. The square root of each MSE agrees with its RMSE to the displayed precision, confirming internal consistency of those two columns.

Table 3. Reported BTC-USD evaluation results. MSE is in USD²; RMSE and MAE are in USD. These scores have not been re-estimated in the revision.

Model	MSE	RMSE	MAE	MAPE (%)
GRU	9,242,998.45	3,040.23	2,366.69	2.7057
STL-GRU	5,085,927.04	2,255.20	1,704.06	2.0791
CEEMDAN-GRU	6,501,786.02	2,549.86	2,189.54	2.5712
Proposed	4,084,400.58	2,020.99	1,674.40	2.0351

Both single-stage decomposition models improve on raw-series GRU in the reported comparison. However, STL-GRU has lower errors than CEEMDAN-GRU. Thus, the results do not support a blanket claim that CEEMDAN is the strongest standalone preprocessing choice. The better ranking of the combined model is consistent with a benefit from selective refinement, although the contribution of the entropy rule itself remains unisolated.

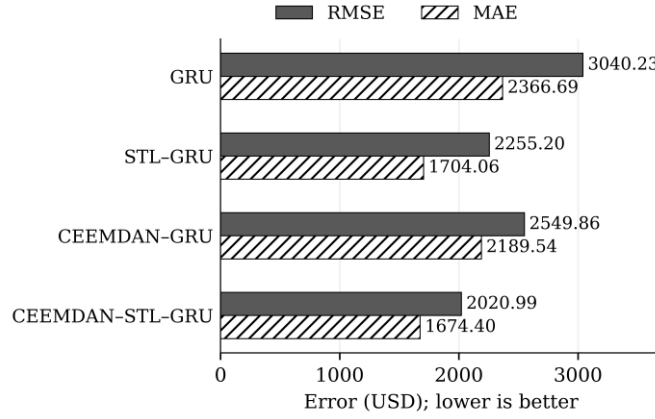


Fig. 2. RMSE and MAE from Table 3, redrawn directly from the reported values. Monochrome fills and hatching distinguish the two metrics; no new predictions are represented.

4.3 Additional Analysis of Relative Gains and Model Count

Table 4 adds a secondary analysis of the reported differences. Each percentage reduction is the baseline error minus the proposed error, divided by the baseline error and multiplied by 100. This calculation requires no additional model fitting and is not an independent validation experiment.

Table 4. Error reductions of the proposed model relative to each baseline, calculated from Table 3.

Baseline	MSE (%)	RMSE (%)	MAE (%)	MAPE (%)
GRU	55.81	33.53	29.25	24.78
STL-GRU	19.69	10.39	1.74	2.12
CEEMDAN-GRU	37.18	20.74	23.53	20.85

Against GRU, RMSE falls by USD 1,019.24 (33.53%) and MAE by USD 692.29 (29.25%). Against CEEMDAN-GRU, the corresponding reductions are 20.74% and 23.53%. The comparison with STL-GRU is more modest: RMSE decreases by USD 234.21 (10.39%), while MAE decreases by only USD 29.66 (1.74%). Because RMSE gives greater weight to large errors, the stronger relative change in RMSE than in MAE suggests that average absolute-error gains are limited; per-date residuals are needed to establish which events explain the difference.

The proposed model uses 11 component GRUs, compared with 9 for CEEMDAN-GRU, an increase of 22.22% in model count. Refining all nine outputs into three STL terms would instead require 27 component models; selective refinement uses 16 fewer, or 59.26% fewer models under that hypothetical design. These are architectural counts, not measured reductions in runtime, memory, or parameter count. The added cost relative to the three-model STL-GRU is particularly relevant given the small MAE gain.

4.4 Limitations and Further Validation

The evidence is limited to one BTC-USD sample, one forecast horizon, and one split. No repeated-seed distribution, confidence interval, statistical forecast comparison, or additional-asset case study is available. The results therefore describe a ranking within this experiment, without establishing statistical significance or generalization across market regimes.

The preprocessing uncertainty in Section 3.5 must be resolved before the reported ranking can support prospective forecasting claims. Chronological partitioning and internally consistent summary metrics cannot substitute for that verification.

Further validation should first recover the original data, dated predictions, and full configuration; then evaluate rolling-origin preprocessing with a persistence baseline and a common tuning budget. Replacing the initial decomposition with EMD, EEMD, VMD, or wavelets would test the choice of CEEMDAN directly. Within the proposed model, comparisons with a fixed IMF, random selection, and refinement of all components would isolate entropy selection. Sensitivity to entropy tolerance, STL period, CEEMDAN noise settings, and random seeds should be examined on development data before an untouched future period is evaluated. Additional assets and horizons would test external validity.

5 Conclusion

We proposed a selective CEEMDAN–STL–GRU framework for one-day-ahead Bitcoin closing-price forecasting. Sample entropy identifies one retained CEEMDAN component for STL refinement, and eleven component GRUs produce an additive price forecast. The reported RMSE is USD 2,020.99, a reduction of 33.53% relative to GRU and 10.39% relative to STL–GRU; the MAE gain over STL–GRU is only 1.74%. CEEMDAN is justified by its adaptive extraction and reconstruction properties, rather than a demonstrated advantage over all competing decompositions. The results motivate further evaluation, subject to verification of causal preprocessing, complete implementation details, and broader statistical and empirical validation.

References

1. Murray, K., Rossi, A., Carraro, D., Visentin, A.: On forecasting cryptocurrency prices: A comparison of machine learning, deep learning, and ensembles. *Forecasting* 5(1), 196–209 (2023). <https://doi.org/10.3390/forecast5010010>
2. Hansun, S., Wicaksana, A., Khaliq, A.Q.M.: Multivariate cryptocurrency prediction: Comparative analysis of three recurrent neural networks approaches. *J. Big Data* 9, 50 (2022). <https://doi.org/10.1186/s40537-022-00601-7>
3. Li, Y., Jiang, S., Li, X., Wang, S.: Hybrid data decomposition-based deep learning for Bitcoin prediction and algorithm trading. *Financ. Innov.* 8, 31 (2022). <https://doi.org/10.1186/s40854-022-00336-7>
4. Chang, T.-J., Lee, T.-S., Yang, C.-T., Lu, C.-J.: A ternary-frequency cryptocurrency price prediction scheme by ensemble of clustering and reconstructing intrinsic mode functions

- based on CEEMDAN. *Expert Syst. Appl.* 233, 121008 (2023). <https://doi.org/10.1016/j.eswa.2023.121008>
5. Dong, Z., Zhou, Y.: A novel hybrid model for financial forecasting based on CEEMDAN-SE and ARIMA-CNN-LSTM. *Mathematics* 12(16), 2434 (2024). <https://doi.org/10.3390/math12162434>
 6. Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y.: Learning phrase representations using RNN encoder–decoder for statistical machine translation. In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar, pp. 1724–1734 (2014). <https://doi.org/10.3115/v1/D14-1179>
 7. Torres, M.E., Colominas, M.A., Schlotthauer, G., Flandrin, P.: A complete ensemble empirical mode decomposition with adaptive noise. In: *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Prague, Czech Republic, pp. 4144–4147 (2011). <https://doi.org/10.1109/ICASSP.2011.5947265>
 8. Huang, N., Shen, Z., Long, S.R., Wu, M.C., Shih, H.H., Zheng, Q., Yen, N., Tung, C., & Liu, H.H.: The empirical mode decomposition and the Hilbert spectrum for nonlinear and non-stationary time series analysis. *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences* 454(1971), 903–995 (1998). <https://doi.org/10.1098/rspa.1998.0193>
 9. Wu, Z., Huang, N.E.: Ensemble empirical mode decomposition: A noise-assisted data analysis method. *Adv. Adapt. Data Anal.* 1(1), 1–41 (2009). <https://doi.org/10.1142/S1793536909000047>
 10. Dragomiretskiy, K., Zosso, D.: Variational mode decomposition. *IEEE Trans. Signal Process.* 62(3), 531–544 (2014). <https://doi.org/10.1109/TSP.2013.2288675>
 11. Mallat, S.G.: A theory for multiresolution signal decomposition: The wavelet representation. *IEEE Trans. Pattern Anal. Mach. Intell.* 11(7), 674–693 (1989). <https://doi.org/10.1109/34.192463>
 12. Stallone, A., Ciccone, A., Materassi, M.: New insights and best practices for the successful use of Empirical Mode Decomposition, Iterative Filtering and derived algorithms. *Sci. Rep.* 10, 15161 (2020). <https://doi.org/10.1038/s41598-020-72193-2>
 13. Richman, J.S., Moorman, J.R.: Physiological time-series analysis using approximate entropy and sample entropy. *Am. J. Physiol. Heart Circ. Physiol.* 278(6), H2039–H2049 (2000). <https://doi.org/10.1152/ajpheart.2000.278.6.H2039>
 14. Cleveland, R.B., Cleveland, W.S., McRae, J.E., Terpenning, I.: STL: A seasonal-trend decomposition procedure based on loess. *J. Off. Stat.* 6(1), 3–33 (1990)
 15. Hyndman, R.J., Koehler, A.B.: Another look at measures of forecast accuracy. *Int. J. Forecast.* 22(4), 679–688 (2006). <https://doi.org/10.1016/j.ijforecast.2006.03.001>